DSP5630xEVM
Host Interface Adapter
Reference Manual

DSP Development Systems

© 1997, SIGNALogic, Inc.
9617 Wendell, Dallas, TX 75243
Tel.: (214) 343-0069 • Fax: (214) 343-0163
Email: dspinfo@signalogic.com
http://www.signalogic.com

Table of Contents

DSP5630xEVM ISA Host Interface 5

PC ISA Interface Card 5

Base I/O Address Selection 6

Active Termination Adaptors 6

ISA Interface Pal Equations 7

ISA Controller Circuit Diagram 8

DSP5630xEVM Active Termination Circuit Diagram 9

ISA Interface I/O Port Equates 10

DSP5630xEVM Printer Port Interface 11

Printer Port I/F Pal Equations 12

Printer Port Interface Circuit Diagram 13

Printer Port Interface I/O Port Equates 14

Optional EVM5630x Board Modification 15

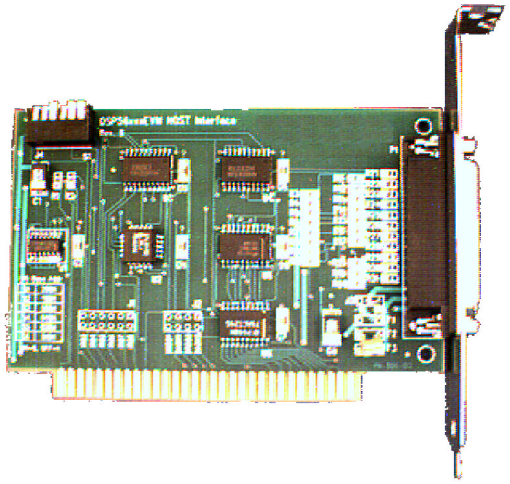
DSP5630xEVM ISA Host Interface

ISA host interface comprises of three parts:

- PC ISA interface card
- Ribbon cable
- DSP5630xEVM active termination

PC ISA Interface Card

An 8-bit ISA controller provides high-speed access to the EVM Host Port. Controller card supplies all power required for EVM operation. To prevent equipment damage both +5VDC and +12VDC are protected with resetable fuses. All control signals are driven through 150 ohm resistors allowing hot connection of the interface to the PC without powering down the host computer.



Base I/O Address Selection

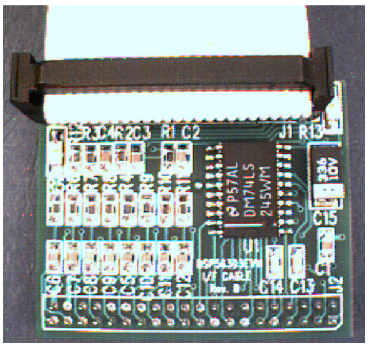
Base I/O address of the ISA interface card is set with six position dip-switch on top of the card. Switch in the ON position sets bit value to 0.

Position	1	2	3	4	5	6
Value	0x200	0x100	0x080	0x040	0x020	0x010

For example, to set base I/O address to 0x240:
switch 1 and 4 are OFF (up)
switch 2, 3, 5 and 6 are ON (down)

Active Termination Adaptors

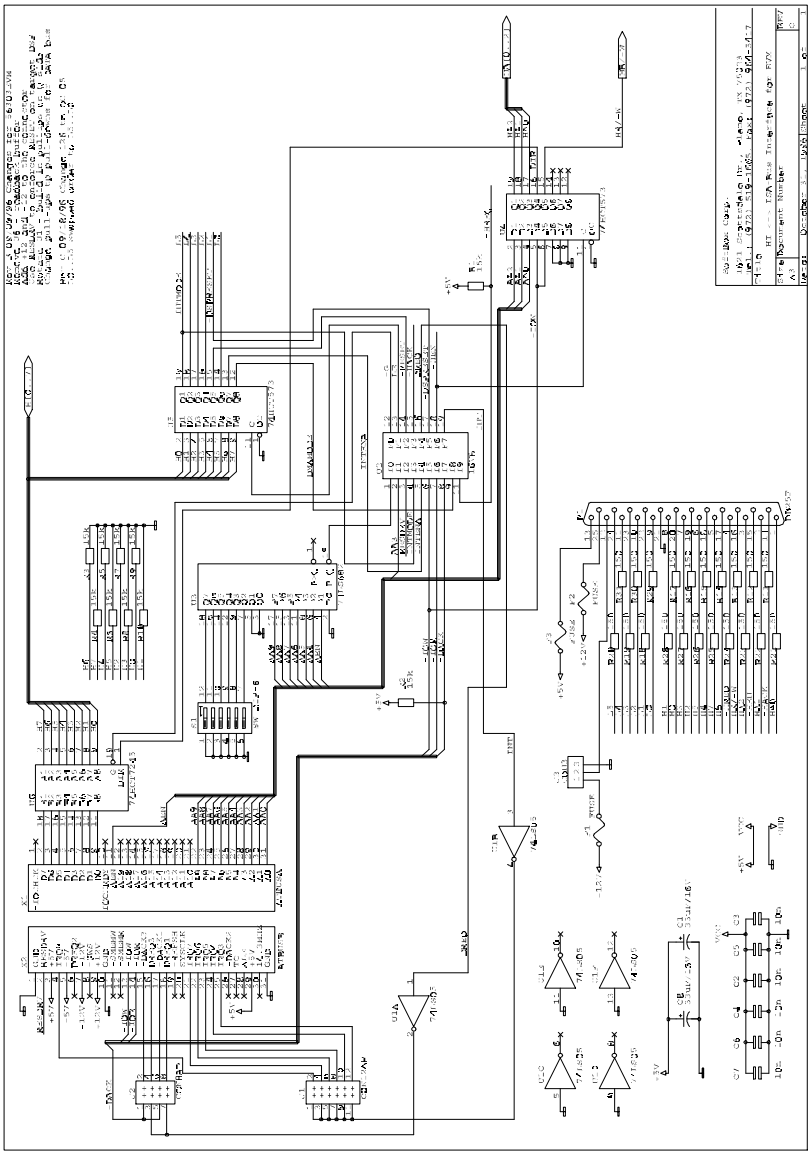
Active termination adaptors connect directly to the Host Port signals of the DSP5630x, so extreme care should be used when connecting or disconnecting the interface. Attaching or disconnecting of the active termination board should be executed with the ribbon cable disconnected from the host PC. Also, external power supply should be disconnected from the EVM, as it will not be needed after connecting the EVM to the PC.



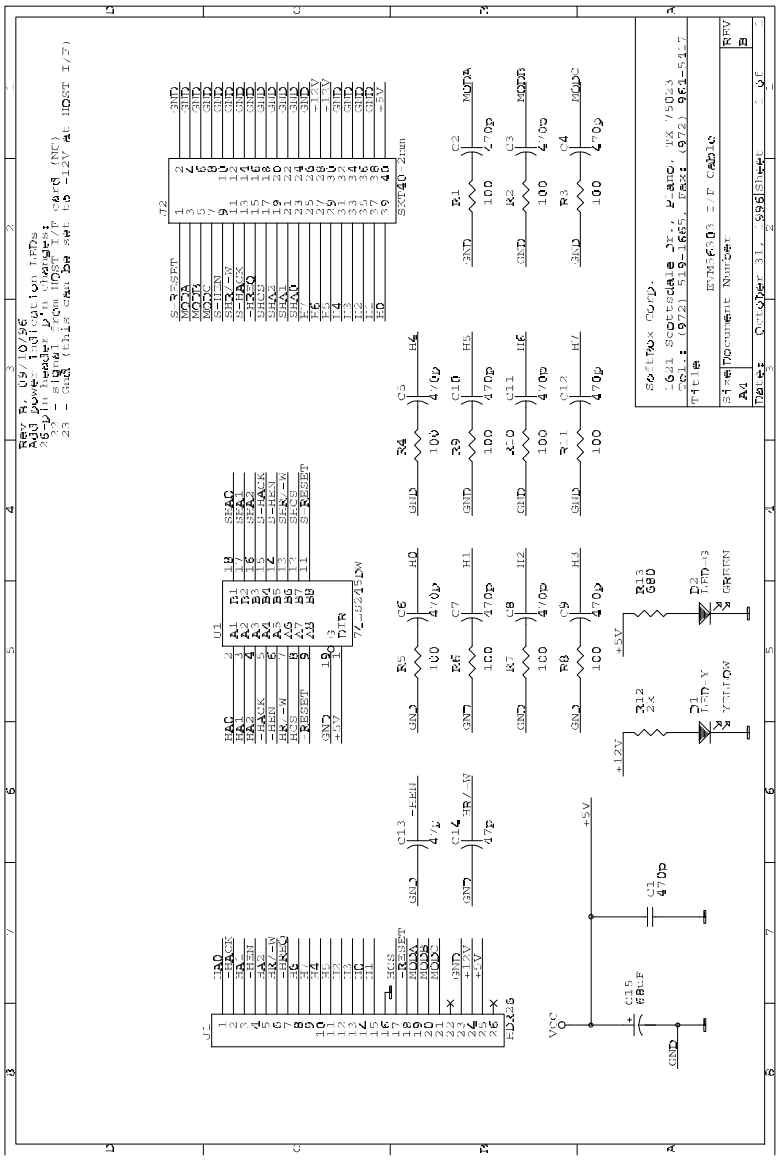
ISA Interface Pal Equations

```
name     HiEvmB_1;
partno   460-001;
date     9/07/96;
revision B;
designer Peter Koenig;
company  SoftBox Corp.;
assembly EvmIsaIF;
location Ux;
device   g16v8;
format   -ja;

PIN [1..5]      = ncard,  aa3,    resdrv, intmode, aa0;
PIN [6..10]     = niow,   nior,   ndack,  dmamode, Gnd;
PIN [11..15]    = nhreq,  ng,     le,     reset,   nhack;
PIN [16..20]    = dreq,   dsprst, nhen,   int,     Vcc;

nhen     = !( !ncard & !aa3 & !niow & nior & ndack
           # !ncard & !aa3 & niow & !nior & ndack );

dreq     = (dmamode & !nhreq );

int      = (!dmamode & !intmode & !nhreq
           # !dmamode & intmode & nhreq);

ng       = !(  !ncard & !aa3 & !niow & nior
           #   !ncard & !aa0 & !niow & nior
           #   !ncard & !aa3 & niow & !nior
           #   !ncard & !aa0 & niow & !nior);

le       = (!ncard & aa3 & !aa0 & !niow & nior
            # resdrv);

dsprst   = !(resdrv # !reset);
```

ISA Controller Circuit Diagram



DSP5630xEVM Active Termination Circuit Diagram



ISA Interface I/O Port Equates

```
#define MODA       0x08
#define MODA       0x04
#define MODA       0x02
#define RESET      0x10
#define HCS        0x20

#define HOST_ICR   (IoBase+0)
#define HOST_CVR   (IoBase+1)
#define HOST_ISR   (IoBase+2)
#define HOST_IVR   (IoBase+3)
#define HOST_HIGH  (IoBase+5)
#define HOST_MID   (IoBase+6)
#define HOST_LOW   (IoBase+7)
#define HOST_CTRL  (IoBase+8)

int resetHost (int IoBase)
{
   IoWrite (HOST_CTRL, MODA+MODB_MODC);       /* Assert reset    */
   Delay (50);                                /* Wait 50 ms      */
   IoWrite (HOST_CTRL, RESET+MODA+MODB_MODC); /* Deassert reset  */
   Delay (50);                                /* Wait 50 ms      */
   if (IoRead (HOST_IVR) != 0x0f)             /* Check IVR value */
      return FALSE;
   return TRUE;
}
```

DSP5630xEVM Printer Port Interface

Printer port interface allows for direct connection to the bi-directional parallel port of IBM PC or compatible computers. This interface allows for moderate speed data transfers between DSP and host PC. Supported are two modes of operation:

- Bi-directional
- EPP Enhanced Printer Port

The “nibble” mode is not supported.

Computers equipped with the ECP interface can be configured as EPP port.

Every access to the DSP’s host port consists of two steps:

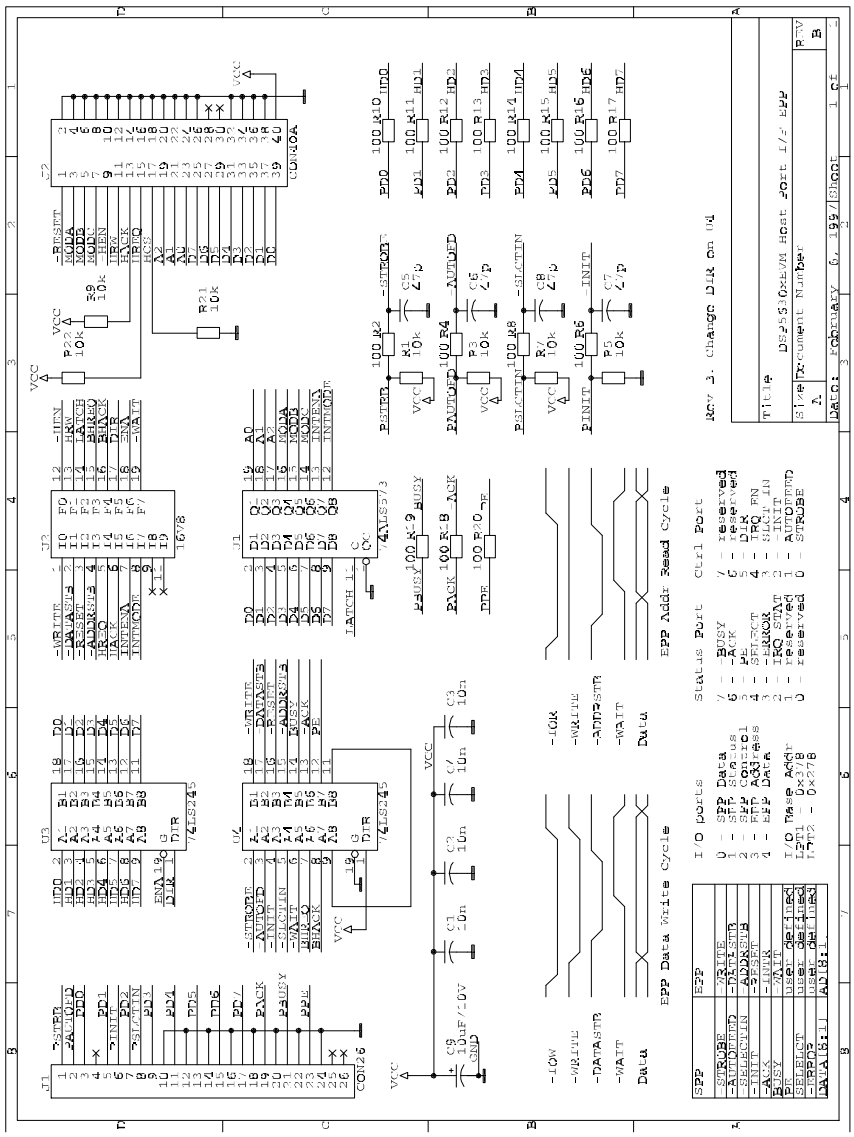
- Write address/control bits
- Read/write data

The printer port interface does not allow +5V power supply to the EVM. The EVM needs to be powered from the standard power supply when the printer port interface is in use.

Printer Port I/F Pal Equations

```
name     HiPP_a;
partno   460-002;
date     10/29/96;
revision A;
designer Peter Koenig;
company  SoftBox Corp.;
assembly PP30xEVM;
location U2;
device   g16v8;
format   -ja;

PIN [1..3]   =                 !WRITE, !DATASTB, !RESET;
PIN [4..8]   = !ADDRSTB, HREQ,  HACK,   INTENA,   INTMODE;
PIN [9..13]  = NC9,      Gnd,   NC11,   !HEN,     HRW;
PIN [14..18] = LATCH,    BHREQ, BHACK,  DIR,      ENA;
PIN [19..20] = WAIT,     Vcc;

DIR   = !WRITE;                  /* Direction of the 245    */

ENA   = DATASTB # ADDRSTB;       /* Enable of the 245       */

LATCH = WRITE & ADDRSTB;         /* Write addr/ctrl bits    */

HEN   = DATASTB;                 /* access host port        */

HRW   = !WRITE;                  /* Read or write host port */

WAIT  = DATASTB # ADDRSTB;       /* handshake with the EPP  */

BHREQ = INTENA & INTMODE & HREQ  /* Interrupt to the PP     */
      # INTENA & !INTMODE & !HREQ;

BHACK = HACK;                    /* status signal to the PP */
```

Printer Port Interface Circuit Diagram



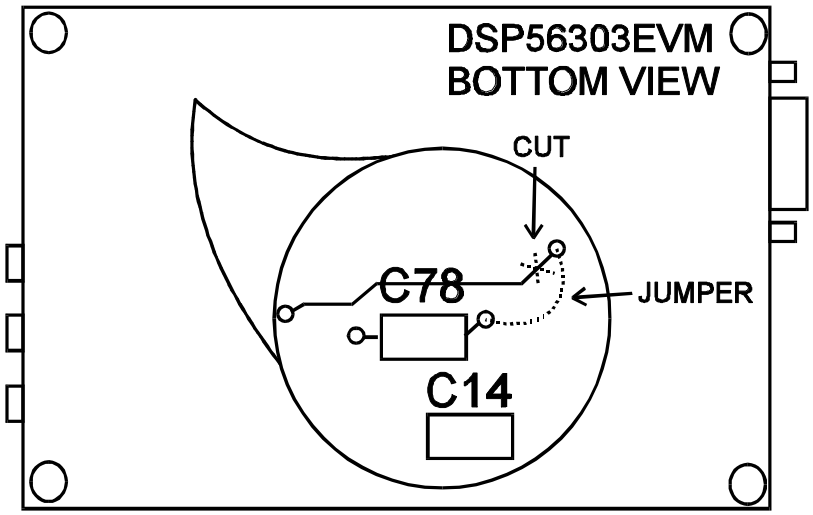
Printer Port Interface I/O Port Equates

```
#define DATA_PORT    (unsigned short)(IoBase + 0)
#define STATUS_PORT  (unsigned short)(IoBase + 1)
#define CTRL_PORT    (unsigned short)(IoBase + 2)
#define EPP_ADDR     (unsigned short)(IoBase + 3)
#define EPP_DATA     (unsigned short)(IoBase + 4)
/* bit definition for the PP control register (CTRL_PORT) */
#define HOST_WRITE   0x01     /* Controls HR/-W and direction of the buffer */
#define DATA_STROBE  0x02     /* Controls -HEN                              */
#define ADDR_STROBE  0x08     /* Controls address LATCH                     */
#define RESET_ON     0x00     /* Reset asserted                             */
#define RESET_OFF    0x04     /* Reset deasserted                           */
#define IRQ_ENA      0x10     /* IRQ to the PC from the PP                  */
#define SPP_READ_ENA 0x20     /* Direction of the PP data buffer (1 - read) */
#define SPP_WRITE    (HOST_WRITE+RESET_OFF)
#define SPP_READ     (SPP_READ_ENA+RESET_OFF)
/* signals on the ADDR latch (PP interface */
#define  HOST_A0     0x01
#define  HOST_A1     0x02
#define  HOST_A2     0x04
#define  HOST_MODA   0x08
#define  HOST_MODB   0x10
#define  HOST_MODC   0x20
#define  INT_ENA     0x40
#define  INT_MODE    0x80
#define  HOST_BOOT   (HOST_MODA+HOST_MODB+HOST_MODC)
#define  HOST_ICR    (0)
#define  HOST_CVR    (HOST_A0)
#define  HOST_ISR    (HOST_A1)
#define  HOST_IVR    (HOST_A0+HOST_A1)
#define  HOST_HIGH   (HOST_A0+HOST_A2)
#define  HOST_MID    (HOST_A1+HOST_A2)
#define  HOST_LOW    (HOST_A0+HOST_A1+HOST_A2)

int resetEpp (int IoBase){
   IoWrite (CTRL_PORT, RESET_ON);              /* Assert reset          */
   IoWrite (EPP_DATA, HOST_IVR); Delay (50);   /* Set Addr Latch to IVR */
   IoWrite (HOST_CTRL, RESET_OFF); Delay (50); /* Deassert reset        */
   return (IoRead (EPP_DATA) == 0x0f)          /* Check IVR value       */
}

int resetSpp (int IoBase){
   int value;
   IoWrite (CTRL_PORT, RESET_ON);              /* Assert reset          */
   IoWrite (DATA_PORT, HOST_IVR); Delay (50);  /* Set Addr Latch to IVR */
   IoWrite (CTRL_PORT, SPP_WRITE); Delay (50); /* Prepare for ADDR write*/
   IoWrite (CTRL_PORT, SPP_WRITE+ADDR_STROBE); /* Write ADDR latch      */
   IoWrite (CTRL_PORT, SPP_WRITE);             /* End strobe ADDR latch */
   IoWrite (CTRL_PORT, SPP_READ);              /* Prepare for read      */
   IoWrite (CTRL_PORT, SPP_READ+ADDR_STROBE);  /* Strobe DATA port      */
   value = IoRead (DATAPORT);                  /* Read IVR              */
   IoWrite (CTRL_PORT, RESET_OFF);             /* End read cycle        */
   return (value == 0x0f)                      /* Check IVR value       */
}
```

Optional EVM5630x Board Modification

Motorola DSP56303 Address Attribute priority logic prevents the onboard 32k x 24 SRAM from being fully utilized as vector (X: and Y:) memory. This feature will be corrected with the next revision of the DSP56303 silicon. To achieve full capability with DSPower and Hypersignal software programs, which require vector memory organization for some functions and instruments, you must make the following modifications to the EVM56303 board:

1) Cut trace that goes under "C78" on underside of EVM56303 board. See diagram below.



2) Jumper the feedthrough (via) at end of "C78" trace to ground. You can use the right-hand pin of C78 to do this.

3) Change jumper JP9 (center-top of component side of EVM56303 board) so that pins 1-2 are connected. On some EVM56303 boards, the default connection may be 2-3, so JP9 should be double-checked.

Note that the DSPower and Hypersignal software demos may display "not enough memory" and other notification messages if the above steps are not implemented.

NOTE ABOUT THE ABOVE CHANGE

After making the above cut-and-trace jumper change, you should AVOID using FLASH memory on the EVM56303, as this will cause contention between the SRAM and FLASH EEPROM devices, and possibly lead to incorrect FLASH programming results or even damage to the the SRAM or FLASH EEPROM devices.